

Cleveland State University
EngagedScholarship@CSU



Electrical Engineering & Computer Science Faculty
Publications

Electrical Engineering & Computer Science
Department

9-2004

Controller Tuning for DC Motor Speed Control Using Genetic Algorithms

Saurabh Jain

Cleveland State University, s.jain1@csuohio.edu

Daniel J. Simon

Cleveland State University, d.j.simon@csuohio.edu

Follow this and additional works at: https://engagedscholarship.csuohio.edu/enece_facpub

 Part of the [Electrical and Computer Engineering Commons](#)

How does access to this work benefit you? Let us know!

Original Citation

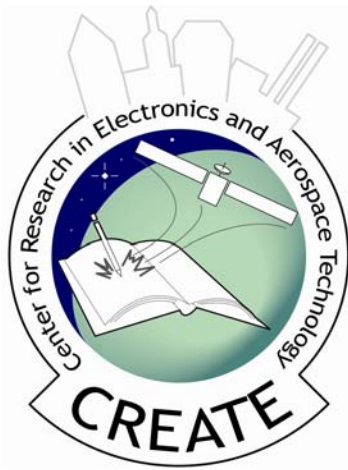
S. Jain and D. Simon, "Controller Tuning for DC Motor Speed Control Using Genetic Algorithms," Aerospace Power and Electronics Simulation Workshop, Cleveland, OH, September 2004

Repository Citation

Jain, Saurabh and Simon, Daniel J., "Controller Tuning for DC Motor Speed Control Using Genetic Algorithms" (2004). *Electrical Engineering & Computer Science Faculty Publications*. 201.

https://engagedscholarship.csuohio.edu/enece_facpub/201

This Presentation is brought to you for free and open access by the Electrical Engineering & Computer Science Department at EngagedScholarship@CSU. It has been accepted for inclusion in Electrical Engineering & Computer Science Faculty Publications by an authorized administrator of EngagedScholarship@CSU. For more information, please contact library.es@csuohio.edu.



Controller Tuning for DC Motor Speed Control Using Genetic Algorithms

Saurabh Jain

Dan Simon

September 21, 2004



Presentation Overview



- Purpose of research
- Solution strategy
- Introduction to Genetic Algorithms (GA's)
 - Concept of GAs
 - Basic Algorithm
 - Representation
- Simplorer and Matlab implementation
- Conclusions and future work



Purpose of research



- To investigate a new method (Diploid GA) of motor control
- To use Simplorer to evaluate design performance
- To consider a broad optimization function for parameter evaluation



Problem Statement



- Speed Control of DC Motors
 - Controller tuning (finding K_p & K_i)
- Aerospace Applications
 - Flywheel energy storage
 - Flight control trim surfaces
 - Hydraulics
 - Fans
 - Thrust vector control
 - Fuel pumps



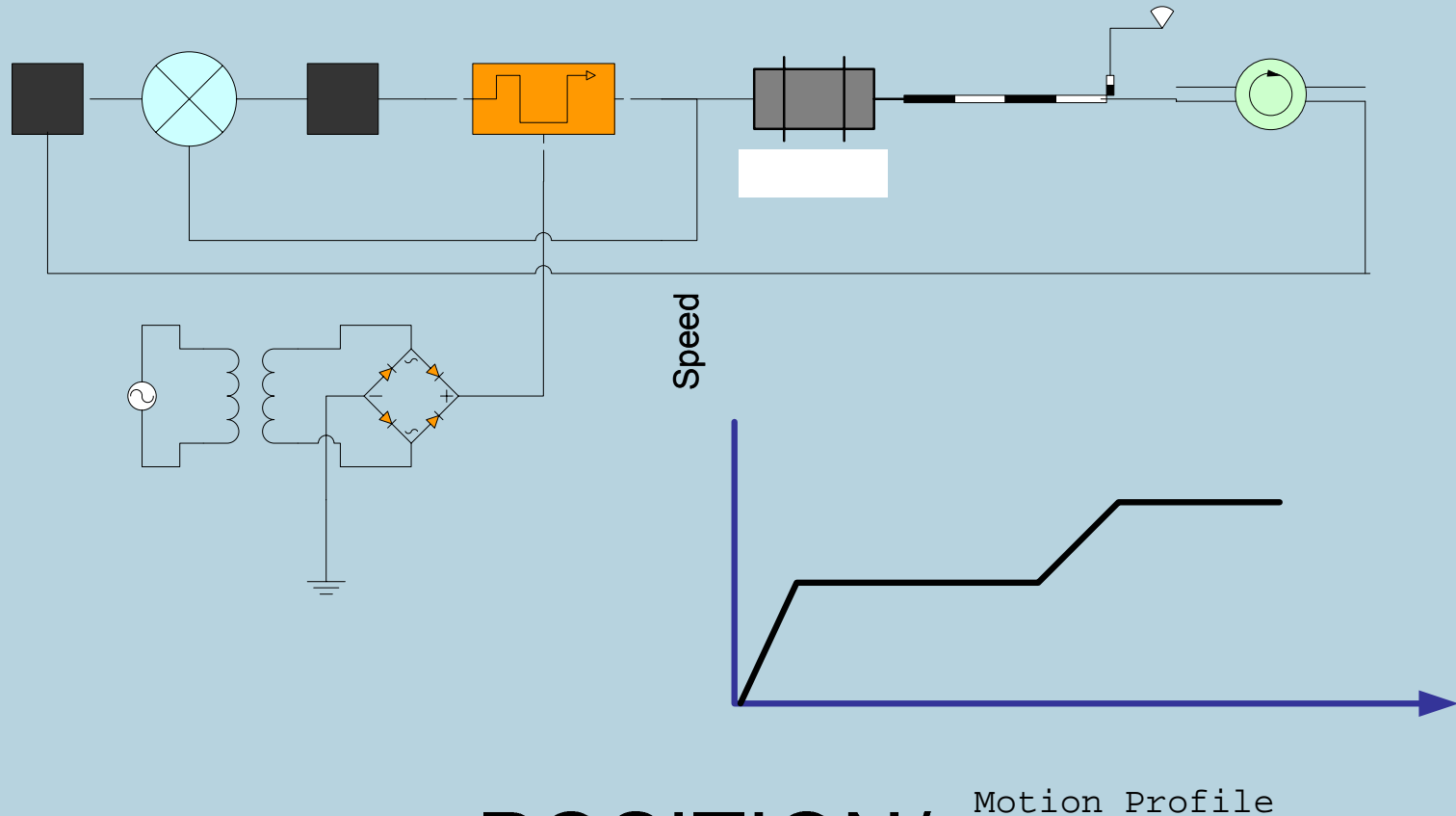
Presentation Overview



- Purpose of research
- Solution strategy
- Introduction to Genetic Algorithms (GA's)
 - Concept of GAs
 - Basic Algorithm
 - Representation
- Simplorer and Matlab implementation
- Conclusions and future work



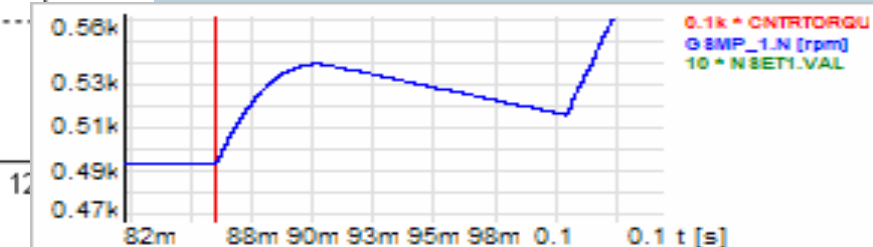
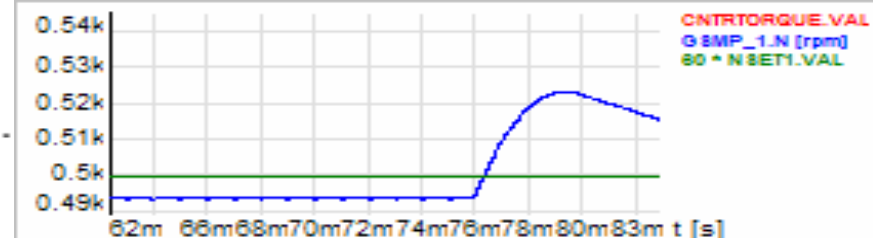
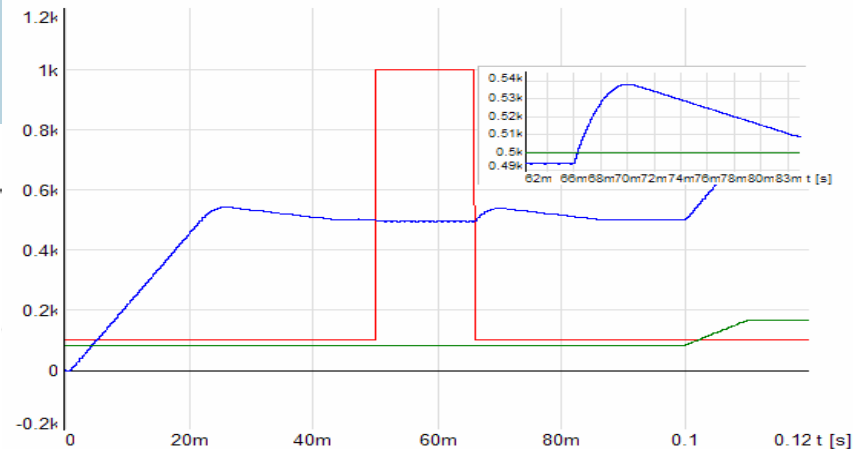
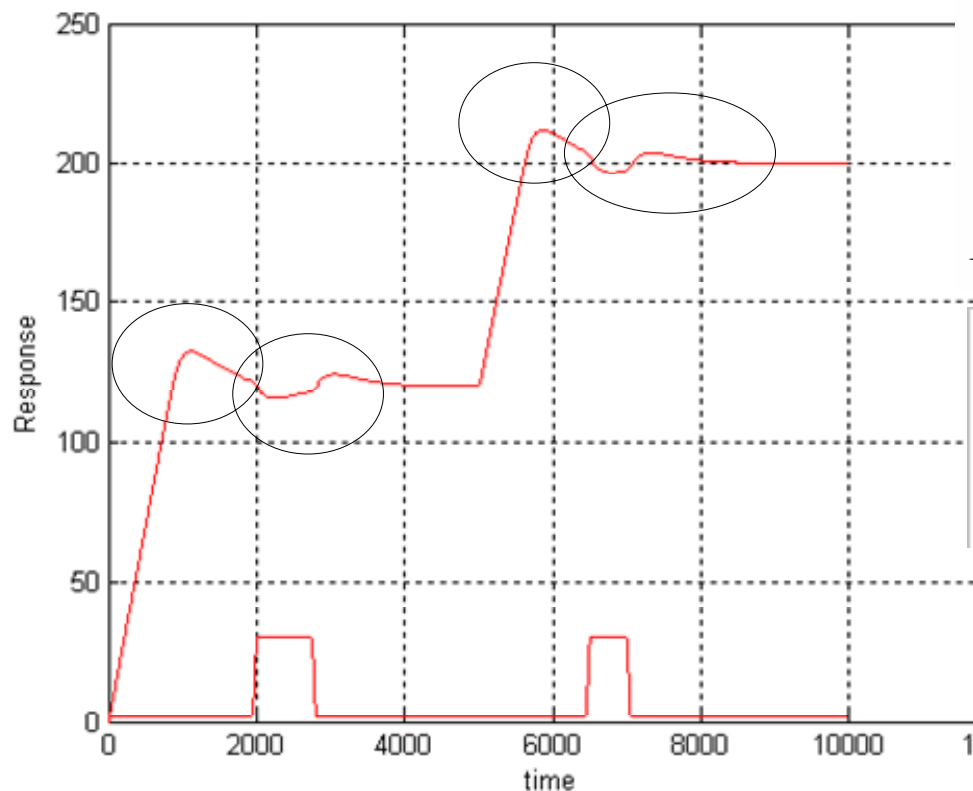
Solution Strategy



POSITION/
SPEED

Optimization function I

Minimize Overshoots



Optimization function II

$$F_s = \phi(i(t), O_m(t))$$

$$F_s = w_1 \max_t i_a(t) + (k_1 O_m|_{s1} + k_2 O_m|_{tc} + k_3 O_m|_{s2} + k_4 O_m|_{tc})$$

$$\min F_s \text{ or } \max 1 / F_s$$

O_m is the maximum overshoot

i_a is the rotor current

k_i are the weighting factors



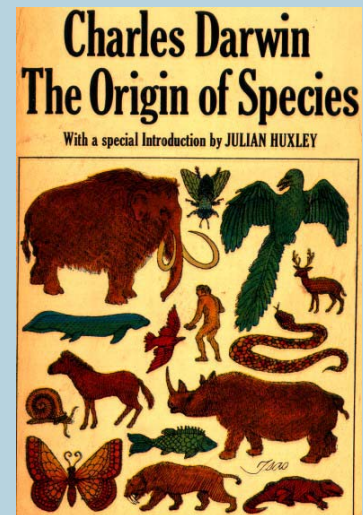
Presentation Overview



- Purpose of research
- Solution strategy
- Introduction to Genetic Algorithms (GA's)
 - Concept of GAs
 - Basic Algorithm
 - Representation
- Simplorer and Matlab implementation
- Conclusions and future work

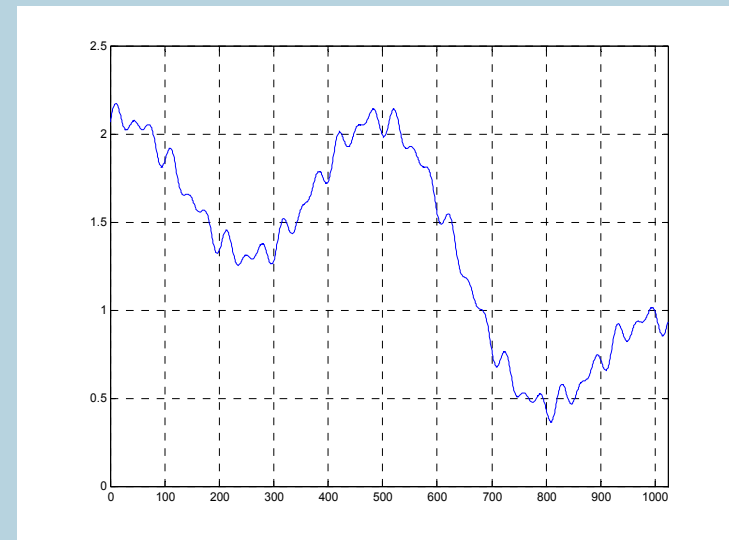
Introduction to GAs

- Genetic algorithms (GA's) are a technique to solve problems which need optimization
- GA's are a subclass of **Evolutionary Computing**
- GA's are based on Darwin's theory of evolution



Concept of GAs I

- Most often one is looking for the best solution in a specific subset of solutions
- This subset is called the **search space** (or state space)
- Every point in the search space is a possible solution
- Therefore every point has a fitness value, depending on the problem definition
- GA's are used to search the search space for the best solution, e.g. a minimum
- Difficulties are the local minima and the starting point of the search





Concept of GAs II



- Starting with a subset of n randomly chosen solutions from the search space (i.e. chromosomes). This is the **population**
- This population is used to produce a next **generation** of individuals by reproduction
- Individuals with higher **fitness** have more chance to reproduce (i.e. natural selection)

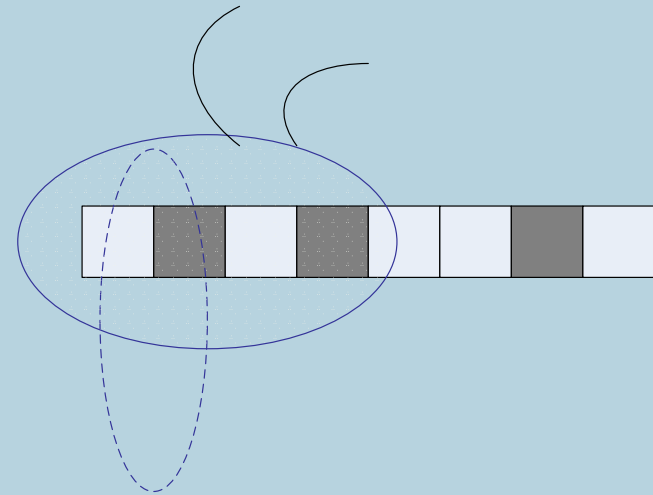


The Basic Algorithm

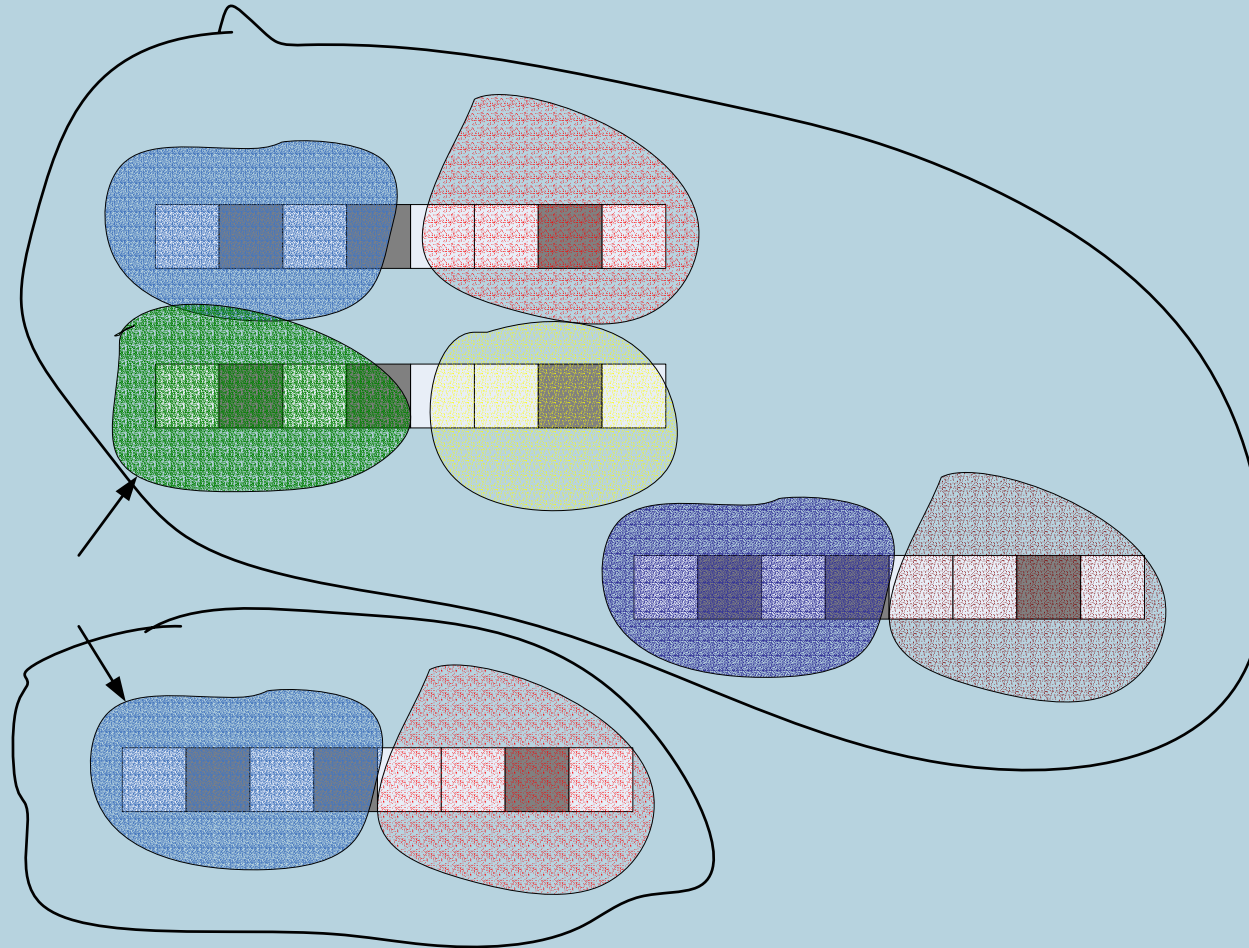
- 0 **START** : Create random population of **n** chromosomes
- 1 **FITNESS** : Evaluate fitness **f(x)** of each chromosome in the population
- 2 **NEW POPULATION**
 - 0 **SELECTION** : Based on **f(x)**
 - 1 **RECOMBINATION** : Cross-over chromosomes
 - 2 **MUTATION** : Mutate chromosomes
 - 3 **ACCEPTATION** : Reject or accept new one
- 3 **REPLACE** : Replace old with new population: the new generation
- 4 **TEST** : Test problem criterium
- 5 **LOOP** : Continue step 1 - 4 until criterium is satisfied

Representation I

- Genetic information is stored in the **chromosomes**
- Each chromosome is build of **DNA**
- The chromosome is divided in parts: **genes**
- Genes code for properties
- The possibilities of the genes for one property is called: **allele**
- Every gene has an unique position on the chromosome: **locus**



Types of GAs



Haploid GAs

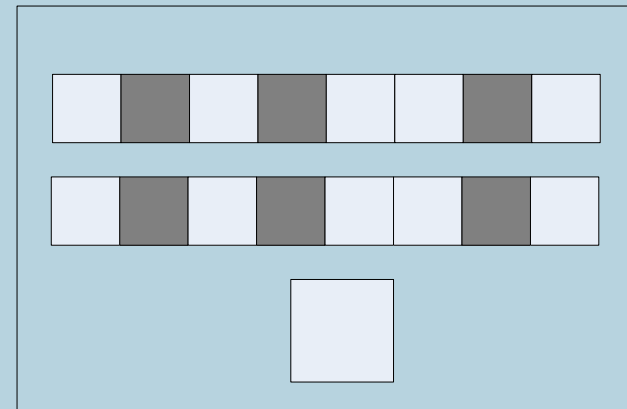
- Easy to use
- Less Computation

Diploid GAs

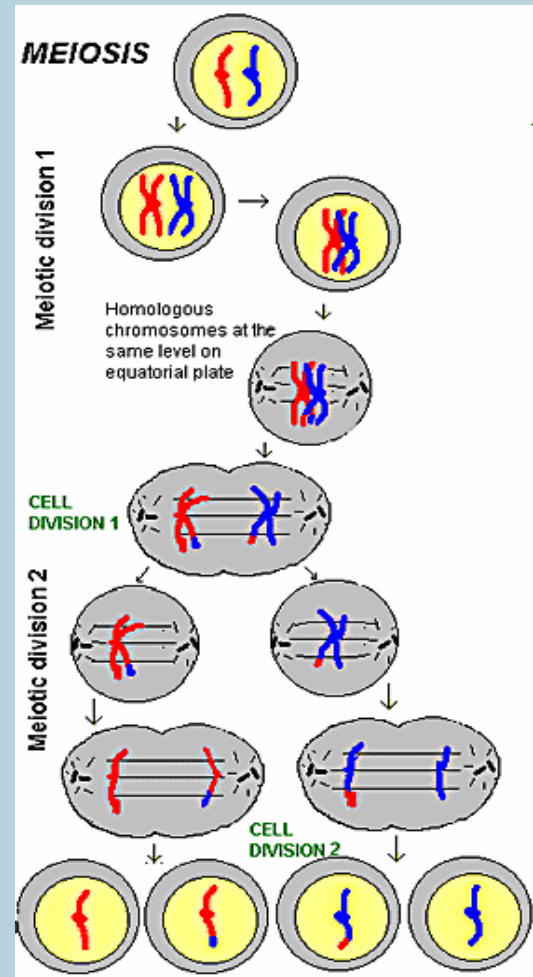
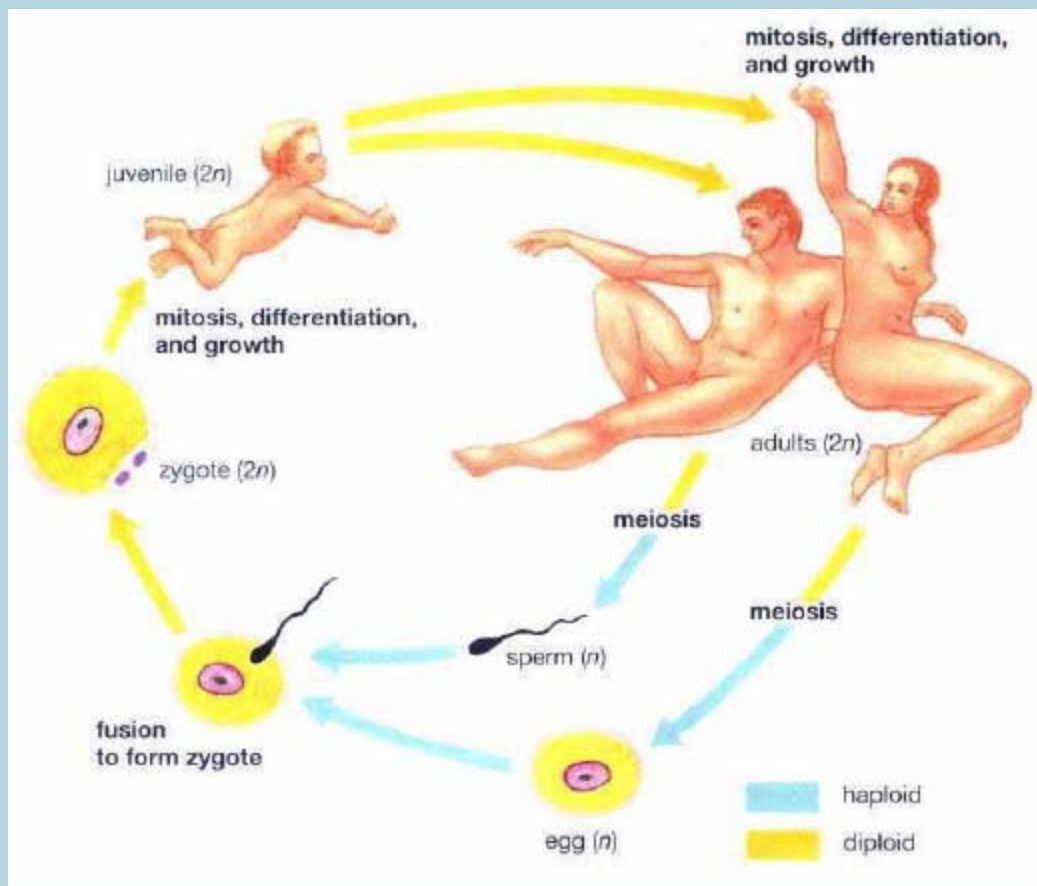
- Good for Dynamic Environments
- More Diversity

Representation II

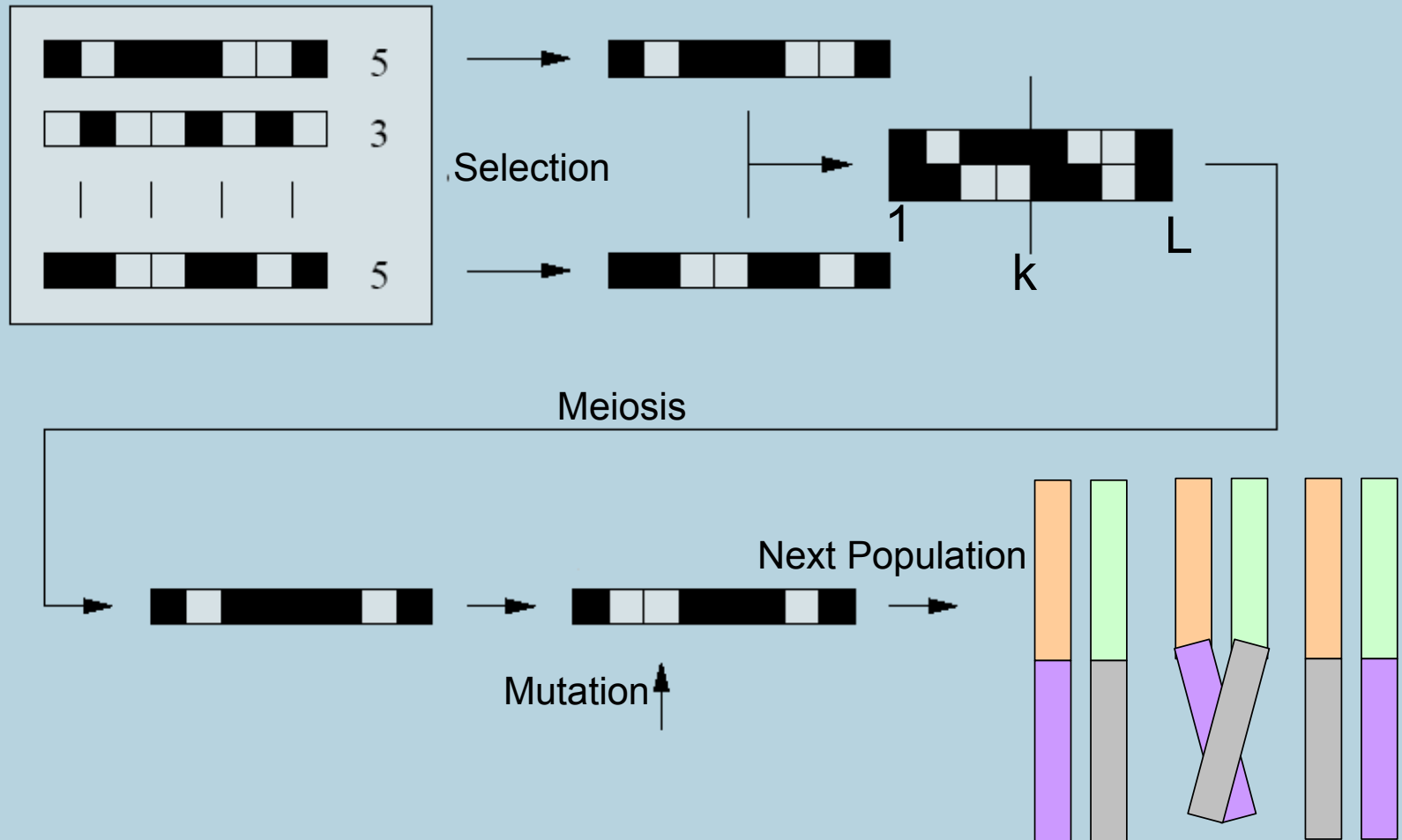
- The entire combination of genes is called genotype
- A genotype develops to a phenotype
- Alleles can be either dominant or recessive
- Dominant alleles will always express from the genotype to the phenotype
- Recessive alleles can survive in the population for many generations, without being expressed



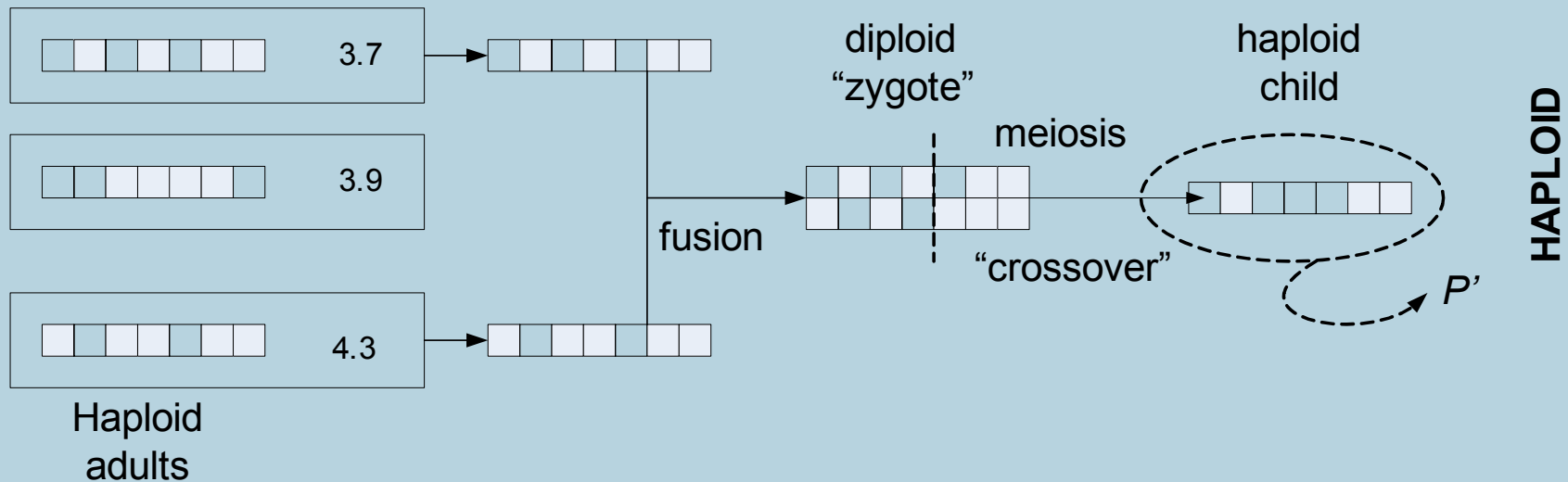
Cell Division



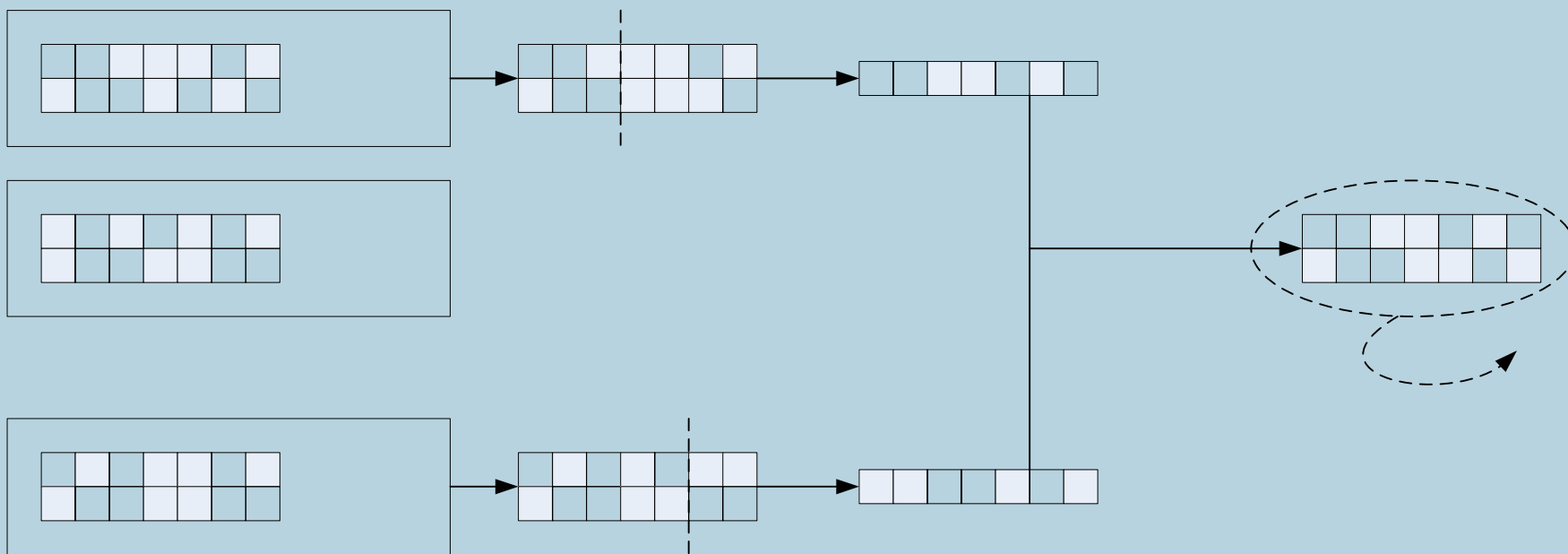
Genetic Operators



Haploid Reproduction



Diploid Reproduction





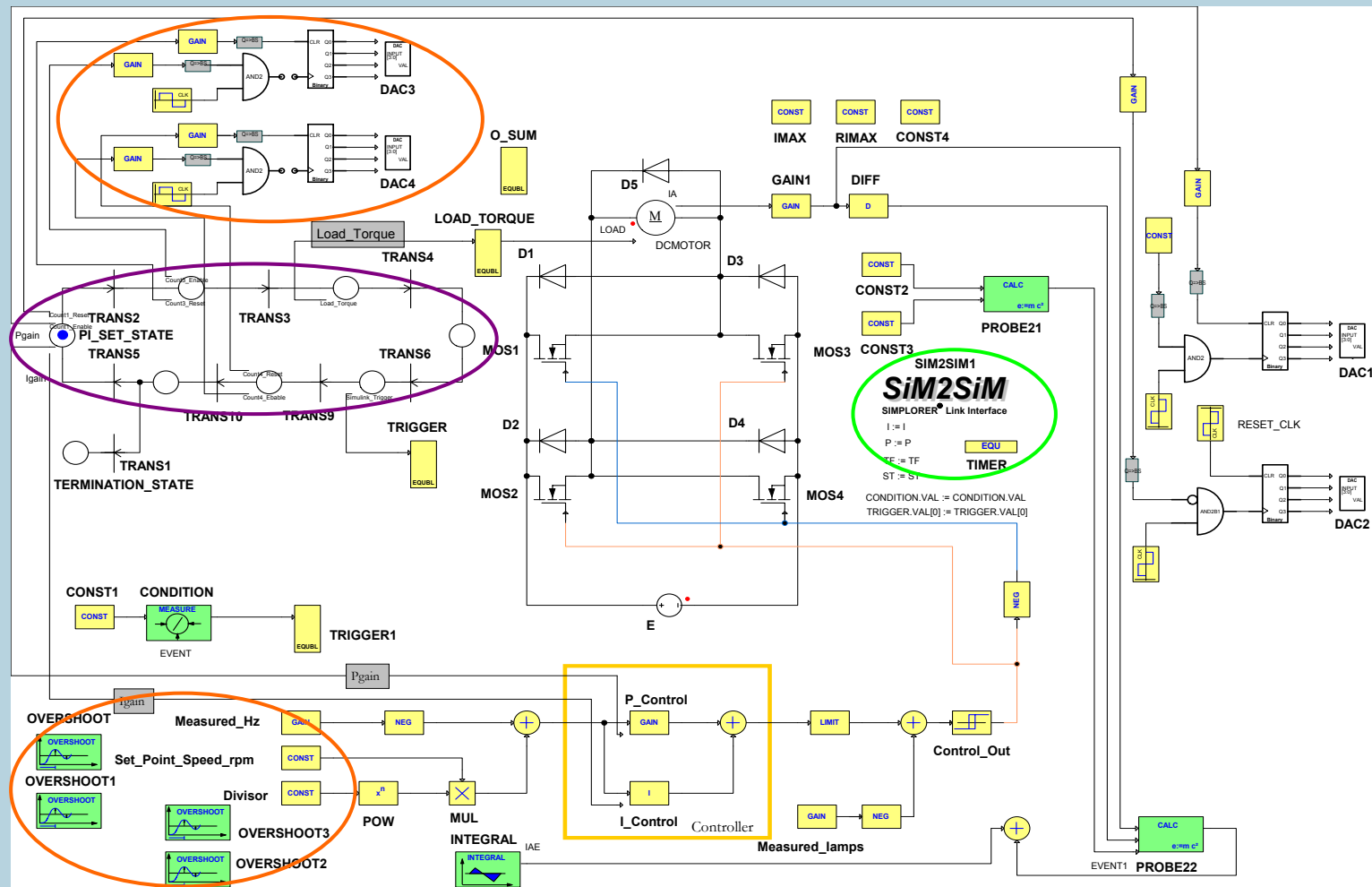
Presentation Overview



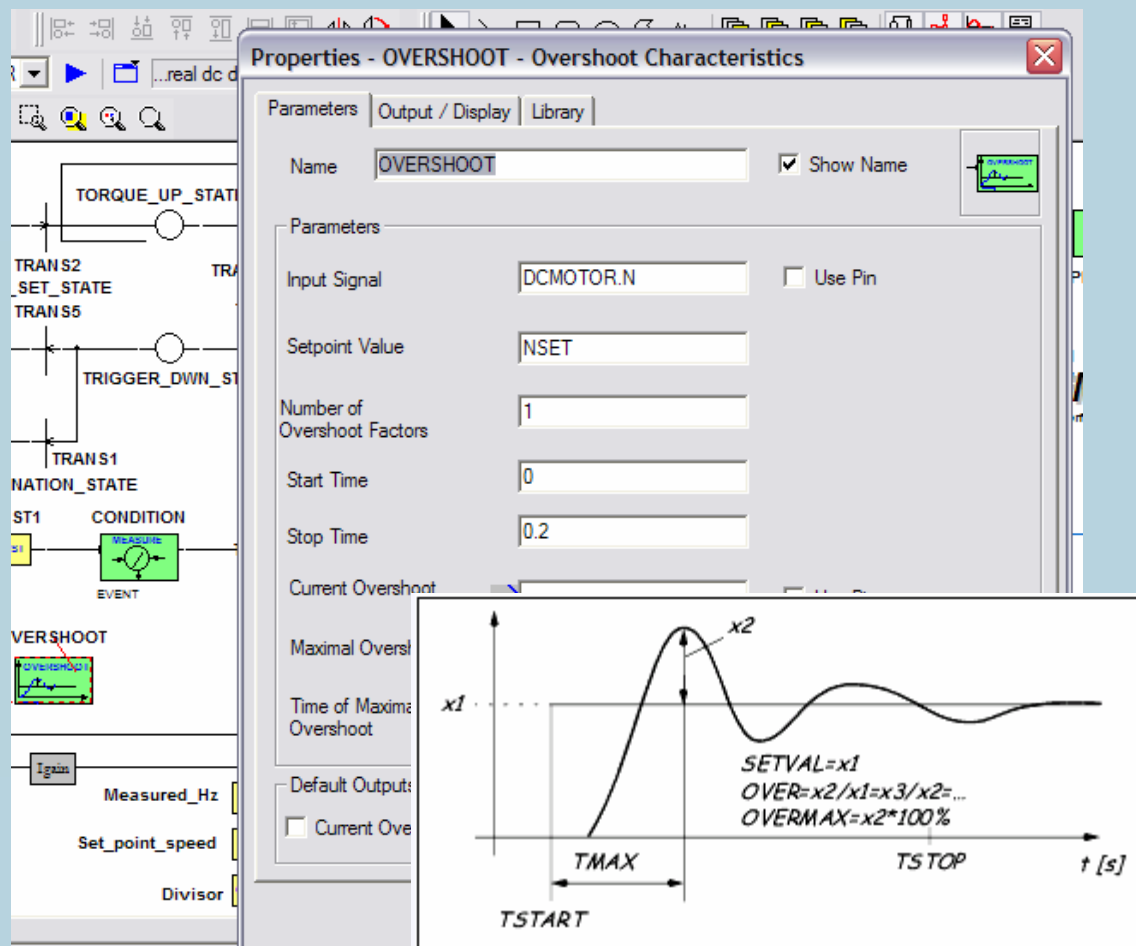
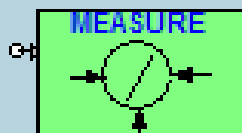
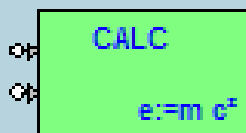
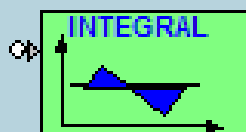
- Purpose of research
- Solution strategy
- Introduction to Genetic Algorithms (GA's)
 - Concept of GAs
 - Basic Algorithm
 - Representation
- Simplorer and Matlab implementation
- Conclusions and future work



Simplorer Model



Functional Blocks



State Flow

Priority 1 = highest

Transition ☐ Show Condition

Logical Expression, Value

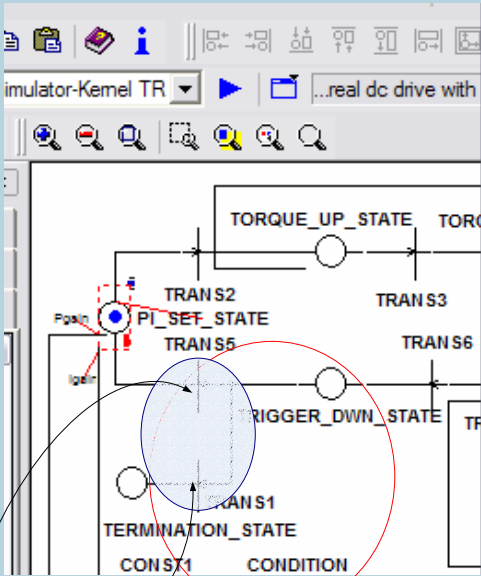


Diagram showing Stateflow transitions and states. Key elements include: TRANS2, TRANS3, TRANS5, TRANS6, TRANS1, TORQUE_UP_STATE, TRIGGER_DWN_STATE, TERMINATION_STATE, and CONST1 CONDITION.

Properties - PI_SET_STATE - State 11

Parameters | Output / Display | Library

Name ☒ Show Name

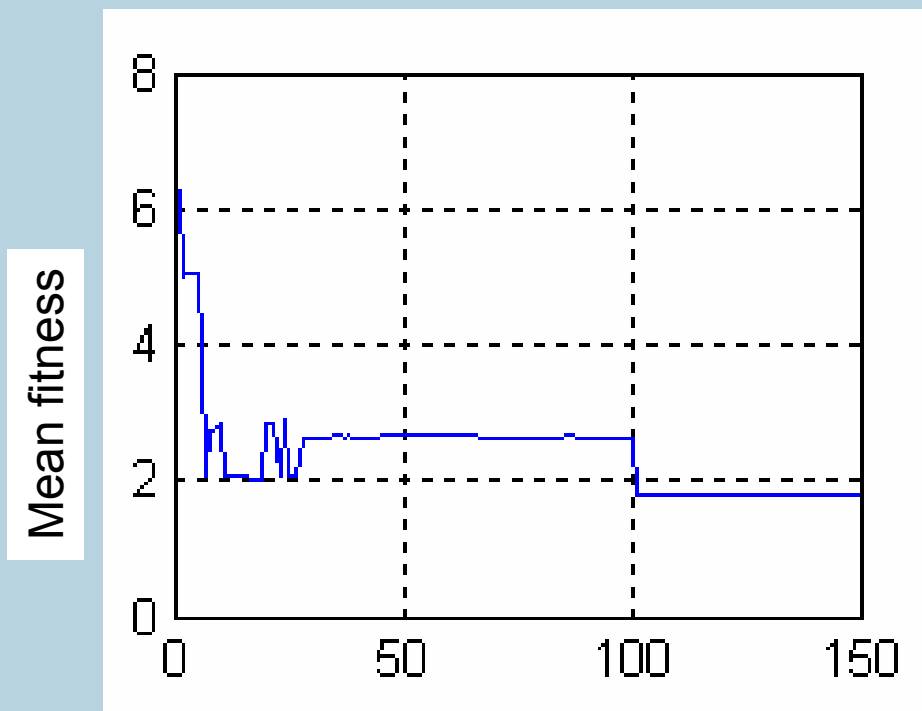
Parameters

Actions ☒ Activate State

Type	Value	Info	Output
STEP:	Pgain:=SIM2SIM1.P		<input type="checkbox"/>
STEP:	Igain:=SIM2SIM1.I		<input type="checkbox"/>
SET:	torque:=0		<input type="checkbox"/>
SET:	Trigger:=0		<input type="checkbox"/>
SET:	Terminat_flag:=SI...		<input type="checkbox"/>



Simulation Results



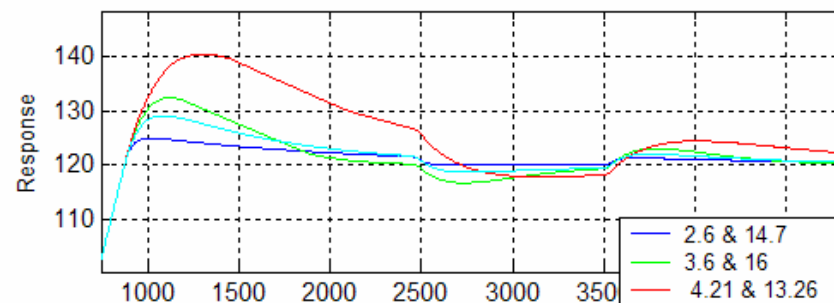
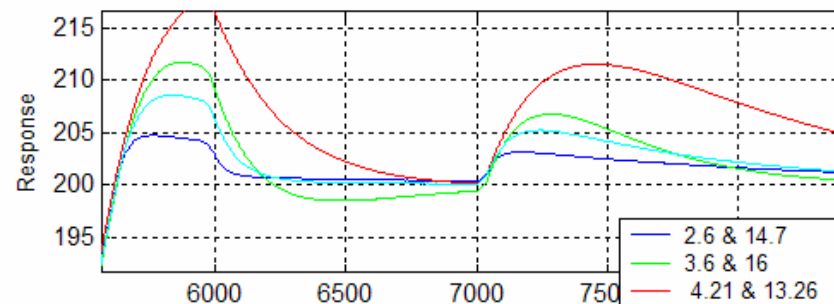
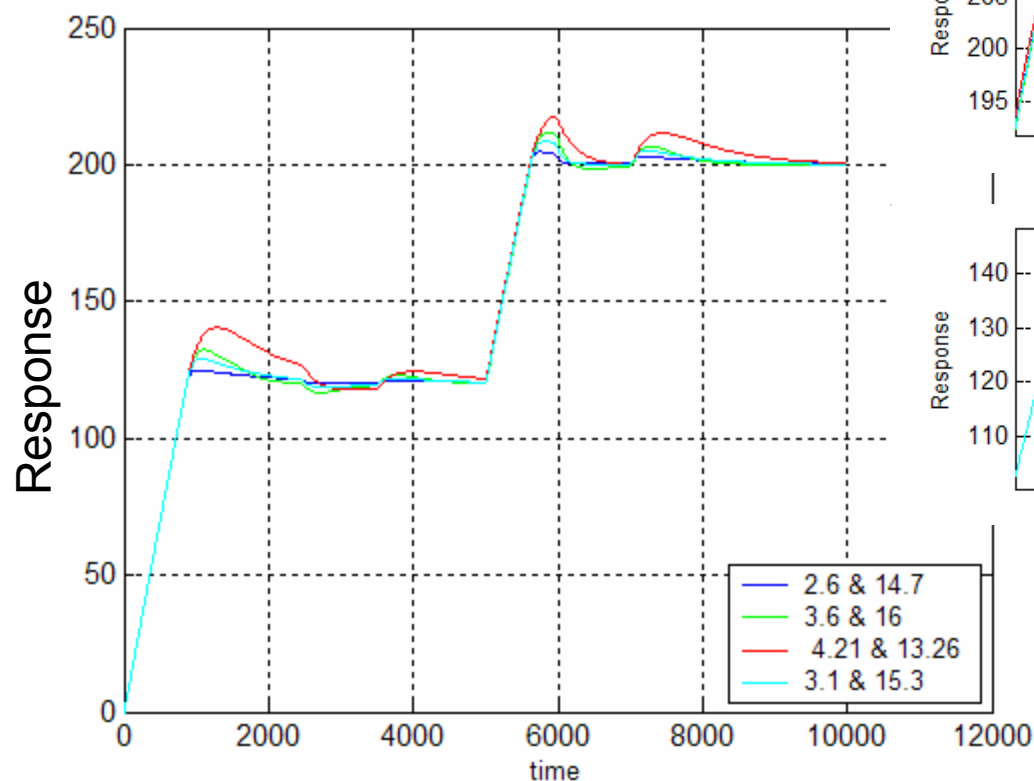
Parameters

Population Size 200

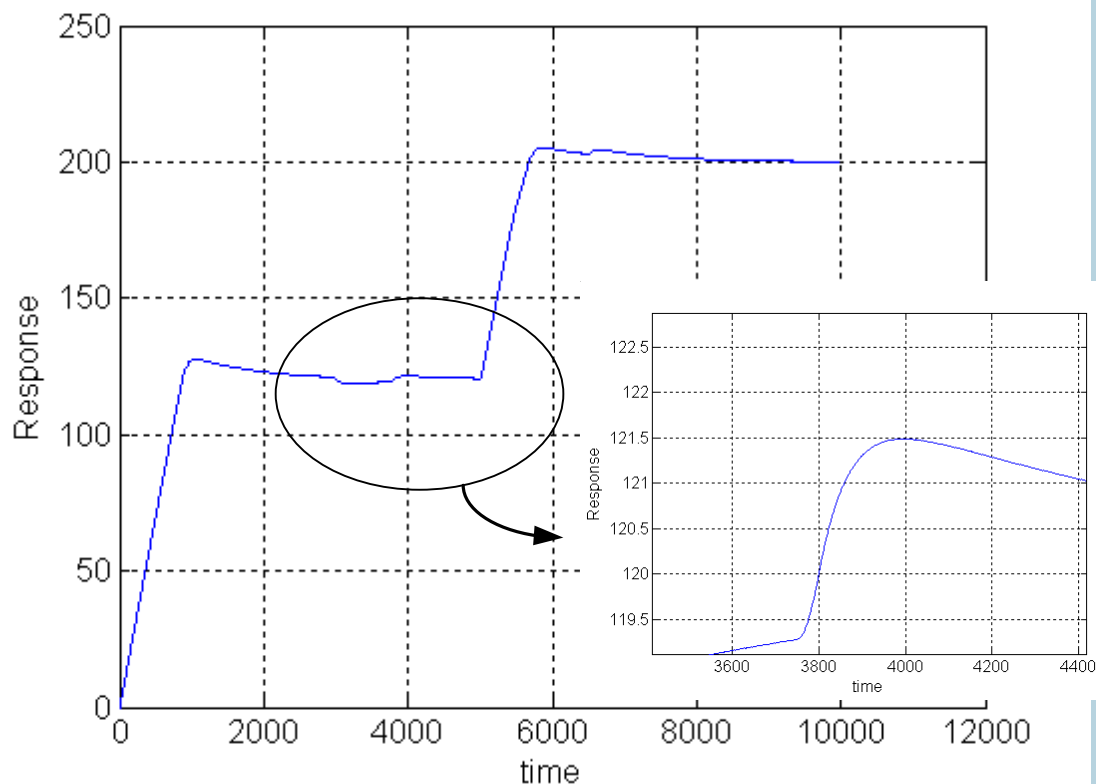
Max Generations 150

Plot of Population Average Fitness and Maximum fitness in each generation vs Number of Generations

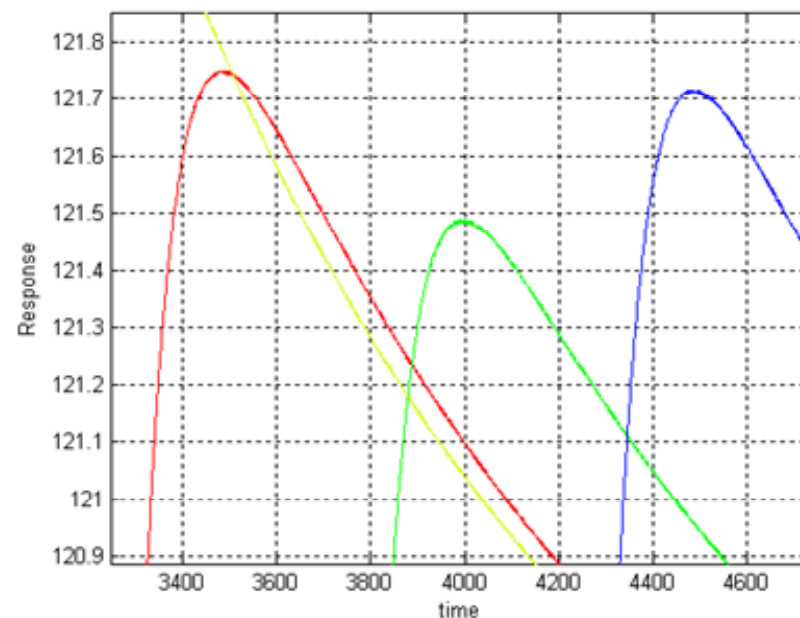
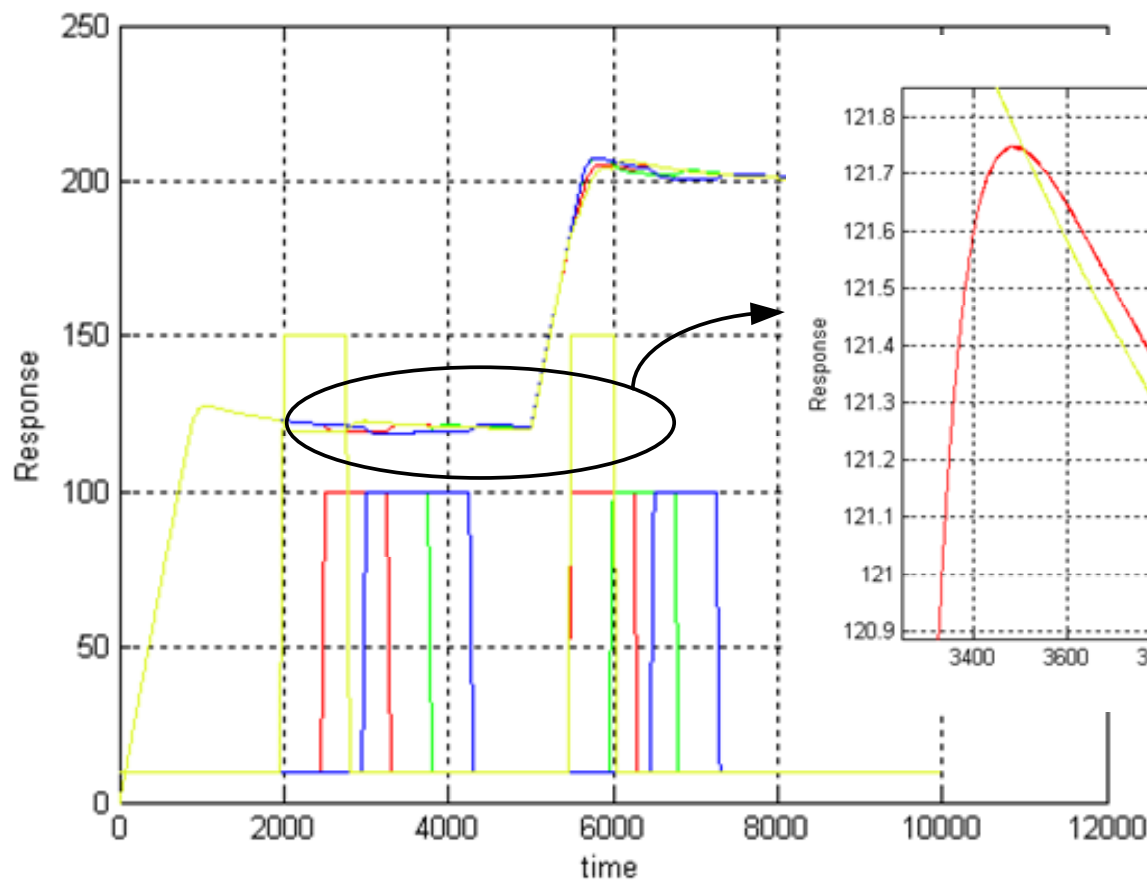
Simulation Results



Simulations Results



Simulation Results





Conclusions and Future work



- Considering the optimization criteria we have achieved satisfactory results. Longer GA runs are expected to perform even better
- This can be expanded to encompass more parameters from the system
- Real time implementation (on-line tuning)
- Comparison with other tuning methods / haploid GA's